Auto-organisation spatio-temporelle : analyse et extension de l'algorithme TOM

Thomas Girod

Auto-organisation spatio-temporelle analyse et extension de l'algorithme TOM

Thomas Girod

11 septembre 2006

Résumé

L'auto-organisation dans les réseaux de neurones est un phénomène aujourd'hui bien maîtrisé dans sa dimension spatiale, généralement par le biais de l'algorithme *Self-Organizing Map* (SOM) de Kohonen [1] et de ses dérivés - près de 4000 articles scientifiques sur le sujet dans la littérature. Mais le pouvoir d'auto-organisation des cartes de Kohonen se limite à des informations spatiales, où les relations entre les individus d'un jeu de données sont simplement ignorées. A l'opposé, le traitement d'informations dynamiques, tel que l'apprentissage de séquences, concentre son apprentissage sur les relations d'ordre qui existent entre les individus d'un jeu de données. Dans la nature, il est rare que l'information soit purement spatiale ou purement temporelle - notre perception de l'environnement, par exemple, se fait toujours dans un cadre temporel continu. Les modèles spatio-temporels visent à réunir les deux informations dans un même processus d'apprentissage. L'algorithme SOM, fort de son succès dans l'apprentissage spatial, a servi de base à l'élaboration de nombreux modèles spatio-temporels. L'algorithme *Time-Organized Map* (TOM) [2], qui est à la base de ce travail, en fait partie. Ce rapport présentera notre travail d'analyse et d'extension de l'algorithme TOM. Plus précisément, nous proposons une vision critique de certains aspects de l'algorithme, des modifications aux aspects critiqués et une extension du modèle aux cartes à deux dimensions.

Table des matières

Première partie

Introduction

Ce document est le rapport d'un stage de Master 2 ayant eu lieu au LORIA (Nancy), au sein de l'équipe Cortex. Ce stage s'est déroulé sous la direction de *Frédéric Alexandre* et *Laurent Bougrain*. Le sujet du stage consistait à étudier les travaux de Wiemer sur l'auto-organisation spatio-temporelle, de faire une analyse détaillée de l'algorithme *Time-Organized Map*, et d'en étudier des extensions possibles.

Le cadre de recherche considéré ici est le domaine des réseaux connexionnistes, et plus précisément des réseaux connexionnistes spatio-temporels (RCST). Le traitement de l'information spatio-temporel est une problématique importante, tout particulièrement dans un domaine bio-inspiré : dans la nature, il est rare que l'information soit purement temporelle ou purement spatiale. A titre d'exemple, la perception sensorielle est un processus continu ou la dimension temporelle a son importance.

En premier lieu, notre travail a consisté à analyser les tenants et aboutissants de la problématique : l'auto-organisation et le traitement de l'information spatio-temporelle. Les RCST sont intéressants et difficiles à mettre en place car ils ont pour but le traitement d'une information double. Il existe des modèles aboutis de l'auto-organisation spatiale ou temporelle, mais le traitement simultané de deux sources d'informations bruitées et intrinsèquement liées rend la tâche bien plus complexe que la simple combinaison d'un modèle spatial et d'un modèle temporel.

En second lieu, nous avons effectué un état de l'art du domaine des RCST, afin de cerner les approches existantes du problème, et de situer l'algorithme TOM parmi celles-ci.

Enfin, nous avons analysé l'algorithme TOM, critiqué certains aspects et proposé des extensions telles que le passage aux cartes à deux dimensions, l'apprentissage séquentiel ou l'utilisation d'une fenêtre temporelle de taille supérieure à un.

Deuxième partie

Travail préliminaire

1 Étude du sujet

Avant toute chose, nous devons bien comprendre le sujet proposé et ses implications. Commençons donc par présenter les concepts fondamentaux : qu'est-ce que l'*auto-organisation* ? Quelle est son expression dans le domaine des réseaux connexionnistes ? Qu'entend-t'on par *spatio-temporel* ?

1.1 Auto-organisation...

Rechercher une définition de l'auto-organisation tend à nous diriger vers le champ de la théorie du chaos. Dans ce domaine, l'auto-organisation est présentée comme étant l'émergence spontanée de structures organisées au sein d'un système, due à l'interaction collective des composants du système.

Les réseaux connexionnistes ont pour but d'extraire de l'information à partir des données présentées itérativement. A chaque présentation, le réseau va modifier sa structure ou ses paramètres par apprentissage pour converger vers un état représentant l'information qu'il aura tiré des données.

Les réseaux connexionnistes sont souvent classés selon le degré de contrôle qu'à l'expérimentateur sur le processus d'apprentissage. On distingue généralement l'apprentissage supervisé, semi-supervisé et non-supervisé. Dans les deux premières catégories, un agent extérieur intervient dans le processus d'apprentissage, afin de guider le réseau vers un état souhaité. La troisième catégorie exclut toute intervention extérieure dans le processus d'apprentissage - le réseau doit donc s'organiser de lui-même. L'état vers lequel le modèle converge est fonction de son état initial (structure du réseau, poids des connexions) et de la loi de probabilité régissant le choix des stimuli présentés à chaque étape.

On retrouve dans ce dernier cas les principes de l'auto-organisation présentés plus tôt : un système constitué de composants dont les interactions entraînent l'émergence d'une structure organisée.

Un intérêt de l'auto-organisation est la souplesse qu'elle procure. Les cartes de Kohonen [1], peuvent aisément s'adapter à des données très différentes. Dans l'idéal, un modèle basé sur l'auto-organisation devrait être capable de s'adapter et d'apprendre à partir de n'importe quelles données. On aurait alors un modèle universel de l'auto-organisation. Le cerveau se base sur l'auto-organisation et a des capacités intéressantes dans le domaine :

- des structures similaires sont utilisées pour coder des informations très différentes. La spécialisation d'une zone s'effectue par adaptation à l'information qui lui est envoyée ;
- l'apprentissage ne fige pas une zone sur un type de traitement. L'organisation peut évoluer avec le temps, en fonction des changements dans les données présentées. Ainsi, on constate que la perte de la vue entraîne une réorganisation progressive des zones qui lui étaient allouées, afin que celles-ci aient un nouveau rôle.

1.2 ...spatio-temporelle

L'auto-organisation dans les RCST est un phénomène qui, comme nous l'avons vu, génère une structure synthétisant l'information présentée au cours du processus d'apprentissage. Le terme *spatio-temporel* fait référence à la nature de l'information apprise par le modèle. Il s'agit de la combinaison de l'information spatiale et de l'information temporelle.

Les données utilisées pour l'apprentissage du modèle sont composées de stimuli, présentés consécutivement au modèle. Le terme *spatial* fait référence au

degré de similitude de forme entre deux stimuli. Cette mesure de similitude se base exclusivement sur les données propres des stimuli (généralement des vecteurs de valeurs numériques), en aucun cas sur les relations particulières qu'ils pourraient entretenir. A l'opposé, les données *temporelles* renseignent les relations qui existent entre les stimuli composant un jeu de données. On fait alors abstraction des considérations spatiales pour exprimer des proximités d'un autre ordre.

Il existe de nombreux travaux sur l'auto-organisation spatiale, qui permettent d'obtenir une topographie guidée par l'information contenue dans les données du jeu d'apprentissage. Dans le champ des réseaux connexionnistes, le modèle d'auto-organisation spatiale le plus connu est certainement l'algorithme *Self-Organizing Map (SOM)* [1]. De même, des modèles de l'organisation temporelle permettent d'obtenir des séquences à partir des stimuli présentés. On pourra citer à titre d'exemple, les modèles de Markov cachés (*Hidden Markov Model*, HMM) [3], qui permettent d'apprendre, à partir d'une séquence bruitée, un graphe d'états représentant cette séquence.

Les jeux de données alliant information spatiale et temporelle sont nombreux. Ainsi, le traitement du signal visuel comprend une composante spatiale - formes, couleurs etc - ainsi qu'une composante temporelle - continuité du phénomène. Or les modèles de l'auto-organisation spatiale ne font aucun cas de l'organisation temporelle des stimuli, et les modèles temporels ne traitent généralement que des séquences de symboles parfaitement reconnaissables. Un modèle spatio-temporel a pour but d'apprendre simultanément des deux sources d'informations.

1.3 Bruit spatio-temporel

Les modèles connexionnistes offrent l'intérêt majeur de permettre un apprentissage sur des données bruitées. Sur des données parfaites - pas d'erreurs, pas d'exceptions - les modèles de l'IA[1] symbolique sont plus adaptés. Dans un modèle spatio-temporel, quelle est la nature du bruit ?

- Le bruit spatial est une altération d'un stimulus. On parle aussi de bruit pour exprimer la diversité des données représentant un même symbole : plus les différents exemples du symbole se ressemblent, moins le symbole est bruité, et inversement.
- Le bruit temporel est une altération des relations qui unissent les stimuli du jeu de données. Ceci peut prendre la forme d'une altération de la séquence (insertion, substitution, suppression) lorsque les stimuli sont liés par des relations d'ordre. Lorsque les relations sont décrites plus finement en pondérant les relations d'ordre par une information temporelle, le bruit peut aussi être une modification de ces écarts temporels.

La coexistence du bruit spatial et du bruit temporel dans un modèle spatio-temporel nous amène à constater qu'il n'est pas possible de faire la distinction entre les deux sources de bruit. A titre d'exemple, une substitution (bruit temporel), peut être interprétée comme un bruit spatial déformant le stimulus au

[1] Intelligence Artificielle

point de le confondre avec un autre. Peut-on alors construire un modèle spatio-temporel en mettant simplement bout à bout un modèle spatial et un modèle temporel ? Si les deux modèles sont capables d'apprendre malgré le bruit, encore faut-il pouvoir faire la différenciation entre les deux sources.

2 État de l'art

Le domaine des Réseaux Connexionnistes Spatio-Temporels (RCST) comprend plusieurs approches du problème de l'apprentissage spatio-temporel. Afin de bien situer la position de l'algorithme TOM, auquel nous nous sommes plus spécifiquement intéressés, au sein des travaux traitant de la même problématique, nous allons présenter le domaine. Plusieurs travaux taxonomiques existent [4, 5, 6, 7, 8, 9], que nous présenterons pour comprendre les grands concepts sur lesquels les RCST existants se basent.

2.1 Taxonomie de Kremer

Dans son travail [4, 5], Kremer présente une généralisation des algorithmes des RCST. Au sein d'un algorithme générique, il met en avant le concept de mémoire à court terme, dont le fonctionnement caractérise les algorithmes des RCST.

Un réseau connexionniste classique possède une mémoire à long terme composée des vecteurs poids de ses neurones. Le mécanisme d'apprentissage à un instant t se base exclusivement sur l'état courant - stimulus présenté et état interne de la mémoire. En revanche, un réseau connexionniste spatio-temporel utilise plus que l'état courant. Il garde le souvenir de son état aux instants précédents dans ce qu'il appelle la mémoire à court terme. C'est sur la manière de gérer cette mémoire à court terme que Kremer base sa taxonomie des RCST.

La taxonomie de Kremer comprend une description très technique et précise des différentes approches, d'un point de vue calculatoire.

2.2 Taxonomie de Gilles Vaucher

Dans sa thèse [6], Vaucher considère que les RCST partent tous sur une base spatiale commune, et que les modifications opérées pour apporter le temps sont à analyser pour distinguer les différentes approches.

Sa première grande subdivision consiste à différencier la prise en compte *extrinsèque* ou *intrinsèque* du temps :

- les modèles à prise en compte extrinsèque n'apportent pas de modification aux méthodes de traitement des données. Au lieu de cela, ils appliquent une modification aux données avant leur traitement afin de leur faire incorporer l'information temporelle ;
- la prise en compte intrinsèque consiste à prendre en compte le temps en apportant des modifications directement dans le réseau.

Dans le cas de la prise en compte intrinsèque, on distingue deux grandes approches :

- la modification de l'architecture, qui consiste à ajouter des connexions qu'on nomme boucles de rétroaction. Ces dernières permettent de faire intervenir les états précédents du réseau dans le processus d'apprentissage ;
- la modification du neurone, qui part du principe que la dimension temporelle peut être gérée localement par le neurone. On a alors des modèles où le neurone calcule son état en fonction de ses entrées présentes et passées.

La taxonomie de Vaucher offre une vision centrée sur la manière d'incorporer l'information temporelle dans un modèle classique.

2.3 Taxonomie de Mozayyani

Mozayyani [7] porte un regard un peu plus large sur le problème en se tournant vers le codage de l'information spatio-temporelle. Il distingue trois méthodes pour coder l'information spatio-temporelle :

- le *codage spatial* consiste à transformer l'information temporelle afin de l'inclure à l'information spatiale. Ceci permet de faire un apprentissage spatio-temporel avec un réseau de neurones classique ;
- le *codage temporel* utilise l'approche inverse : il consiste à coder l'information spatiale sous une forme temporelle [10], en faisant varier le la latence de l'émission en fonction du potentiel d'action ;
- avec le *codage spatio-temporel*, l'information spatiale et l'information temporelle sont toutes deux présentes dans le flux de données sous leur forme originelle. Dans ce cas, il est bien sur nécessaire que le réseau soit apte à traiter les deux sources d'informations en simultané ;

Dans le cas d'un codage spatio-temporel, Mozayyani distingue deux cas que l'on retrouve dans la taxonomie de Gilles Vaucher : le *codage global* et le *codage local*.

- le codage global intègre le temps au niveau de l'architecture et de l'algorithme d'apprentissage,
- tandis que le codage local intègre le temps au niveau du fonctionnement du neurone.

2.4 Taxonomie de Zouhour Neji Ben Salem

Dans son travail de synthèse, Neji Ben Salem [8] retient une taxonomie qui reprend en partie les taxonomies de Mozayyani et de Vaucher. Elle se base sur deux points de vues selon lesquels les modèles RCST sont observés :

- le premier fait le lien entre le codage extrinsèque de Vaucher [6] et les trois types de codage de l'information spatio-temporelle énoncés par Mozayyani [7]. La forme de l'information avant son traitement par le réseau de neurones est classée ici ;
- le deuxième regroupe trois approches pour le codage intrinsèque : la modification de l'architecture, la modification du neurone, et la modification du traitement.

2.5 Position de l'algorithme TOM

Après lecture de ces taxonomies, nous pouvons voir les caractéristiques particulières de l'algorithme TOM, permettant ainsi de le positionner dans le domaine.

- L'algorithme TOM traite une information spatio-temporelle au sens de Mozayyani : les données contiennent l'information spatiale et l'information temporelle sous deux formes distinctes ; aucun traitement n'est effectué sur les données avant leur entrée dans le réseau - il s'agit donc d'un traitement intrinsèque au sens de Vaucher.
- Le traitement interne de l'information temporelle se fait au niveau de l'algorithme, plutôt qu'au niveau du neurone ; il s'agit d'un modèle à traitement global au sens de Mozayyani.

3 Inspirations et Antécédents

3.1 Le modèle SOM

L'algorithme TOM est présenté comme une extension de l'algorithme SOM, visant à ajouter le traitement de la dimension temporelle des données. Comme de nombreux modèles étendant les travaux de Kohonen, l'algorithme TOM reprend les concepts de base de SOM, que nous présenterons ici.

Tout d'abord, l'architecture se base sur des cartes de neurones. On considère qu'une carte 2D est une bonne approximation de l'organisation des neurones au sein du cortex [11].

Le réseau se base sur deux couches : la première est composée de senseurs et la deuxième de neurones de traitement. Chaque neurone de traitement n_i possède un vecteur de poids w_{n_i} qui représente ses connexions avec la couche de senseurs. On se contente généralement d'un cas simple, dans lequel chaque neurone de traitement est connecté à toute la couche de senseurs.

La procédure effectuée à chaque pas de temps est la suivante :

- sélectionner aléatoirement un stimulus du jeu de données s selon une loi de probabilité définie ;
- présenter ce stimulus au réseau par le biais de la couche de senseurs ;
- pour chaque neurone n_i de la carte, calculer $d_i = |s - w_{n_i}|$, la distance du neurone au stimulus présenté. La mesure de distance peut varier suivant les implantation - ici, nous prendrons une simple distance euclidienne ;
- $min(d_i), \forall i$ nous permet de définir le neurone vainqueur n_{ff}, le plus proche du stimulus présenté. Par la suite nous nommerons cette étape de détermination du vainqueur le *feedforward* ;
- on adapte le vecteur poids du vainqueur pour qu'il s'approche du stimulus présenté : $w_{n_{ff}} = w_{n_{ff}} + (s - w_{n_{ff}}) * \alpha$, où α est le taux d'apprentissage ;
- on fait apprendre les voisins de n_{ff} avec un taux d'apprentissage moindre. La détermination du taux d'apprentissage des neurones voisins se fait en suivant une gaussienne.

En répétant cette étape un grand nombre de fois et en tirant à chaque fois un stimulus selon une loi de probabilité donnée, on obtiendra une carte dont les vecteurs poids des neurones seront représentatifs du jeu d'apprentissage, et dont la topographie fera en sorte que des neurones voisins aient des vecteurs poids aussi semblables que possible.

3.2 De SOM à TIS

Dans ses travaux antérieurs [11] au développement de l'algorithme TOM, Wiemer pose les bases de son approche de l'apprentissage spatio-temporel. Son but est de donner une influence à l'information temporelle dans le processus d'auto-organisation des cartes de neurones. Cette influence passer par une transposition des écarts temporels entre les stimuli en distances spatiales sur la carte. Ainsi :

> Common models for stimulus-induced learning within topographic representations are based on the stimuli's spatial structure and probability distribution. Furthermore, we argue that average temporal stimulus distances reflect the stimuli's relatedness. As topographic representations reflect the stimuli's relatedness, the temporal structure of incoming stimuli is important for the learning in cortical maps.

Ici, Wiemer aborde son idée première : dans le cadre d'un jeu de données où les stimuli entretiennent des relations temporelles, la distance temporelle séparant deux stimuli contient une information sur leur proximité spatiale. Les stimuli sensoriels en sont un bon exemple : ils possèdent une dynamique continue, au sein de laquelle le stimulus évolue progressivement, et la manière dont l'évolution se produit a un sens. Ainsi, sur le cas de la perception visuelle, deux images perçues consécutivement seront généralement similaires, ou un changement brusque entre deux images proches temporellement aura une signification.

C'est à partir de cette considération que Wiemer développe le modèle *Temporal Integration Segregation* (TIS). Le but est de construire un modèle qui puisse simuler certains phénomènes que les modèles classiques d'auto-organisation spatiale ne sont pas capables d'expliquer. Les arguments principaux sur lesquels TIS se base sont les suivants :

- l'expérience neurobiologique de Spengler et al. [12] sur la plasticité du cortex sensoriel primaire dénotent un phénomène d'intégration des stimuli synchrones, et de ségrégation des stimuli asynchrones. En entraînant des singes à distinguer des stimuli tactiles partiellement superposés spatialement, ils mettent en évidence deux phénomènes dans le l'hémisphère expérimental : une zone réagissant à des stimuli synchrones mais spatialement dispersés se forme (phénomène d'intégration), et des stimuli similaires espacés dans le temps génèrent des zones dispersées réagissant à ceux-ci (phénomène de ségrégation) ;
- la localisation spatiale de plusieurs stimuli est faussée lorsque ceux-ci sont présentés à des intervalles temporels inférieurs à quelques centaines de mil-

lisecondes. Ce phénomène, appelé *saltation* [13], illustre la transformation des écarts temporels en distances spatiales perçues ;
- l'apprentissage hebbien est un candidat probable pour être le fondement de la plasticité corticale ;
- les stimulations naturelles sont continues ; par conséquent, la proximité temporelle génère une topologie exprimant des similarités entre les stimuli.

Le modèle TIS permet de stimuler ces phénomènes tout en gardant à l'esprit la crédibilité biologique. Pour cela, il se base sur un modèle distribué, basé sur l'apprentissage hebbien, composé de trois cartes de neurones à deux dimensions (fig. 1) :
- une carte de neurones senseurs,
- une carte intermédiaire où chaque neurone (i, j) est connecté au neurone sensoriel (i, j) correspondant,
- une troisième carte de traitement où chaque neurone est connecté à tous les neurones de la carte précédente.

Dans ce modèle, les stimuli forment une séquence et l'information temporelle prend la forme d'intervalles temporels inter-stimuli (*Inter Stimuli Interval*, ISI). Les stimuli sont donc présentés à intervalles de temps discrets en suivant cette information temporelle, mais le modèle en lui-même est continu et poursuit ses opérations entre deux stimuli. Lorsqu'un stimulus est présenté au modèle :
- l'activité générée est transmise à la carte intermédiaire, qui est excitée en conséquence. Cette excitation vient s'ajouter à l'état courant de la carte. Le rôle de cette carte est de faire décroître son activité progressivement. Ainsi, plus deux stimuli présentés successivement sont proches temporellement, plus ils apparaîtront comme ne formant qu'un seul stimulus au sein de la carte intermédiaire ;
- l'activité de la carte de traitement est la somme de son état courant et des activations provoquées par la carte intermédiaire. Les poids des connexions entre ces deux couches sont adaptés selon un apprentissage hebbien. L'excitation de la carte de traitement évolue de manière continue, mais selon un schéma différent de la carte intermédiaire. Ici, le potentiel d'excitation est diffusé vers les voisins, provoquant une propagation en vague. Ici se trouve le point clef de la méthode de Wiemer : la transposition du temporel vers le spatial se fait à l'aide d'une vague qui se propage à vitesse constante pendant un temps donné. Lorsqu'un stimulus est présenté, l'activité provoquée est localisée à l'aide de connexions inhibitrices récurrentes. La vague présente sur la carte lors de la présentation d'un nouveau stimulus provoque un décalage de l'activité provoquée par le nouveau stimulus.

3.3 De TIS à TOM

Avec le modèle TIS, Wiemer présente une méthode pour transposer l'information temporelle vers le spatial, afin de pouvoir la coder dans la topographie de la carte. Cette conversion temps - distance par l'usage d'une propagation de potentiel en vague permet de simuler des phénomènes observés dans le vivant. En cela, le modèle TIS rempli ses objectifs. Cependant, le modèle TIS est

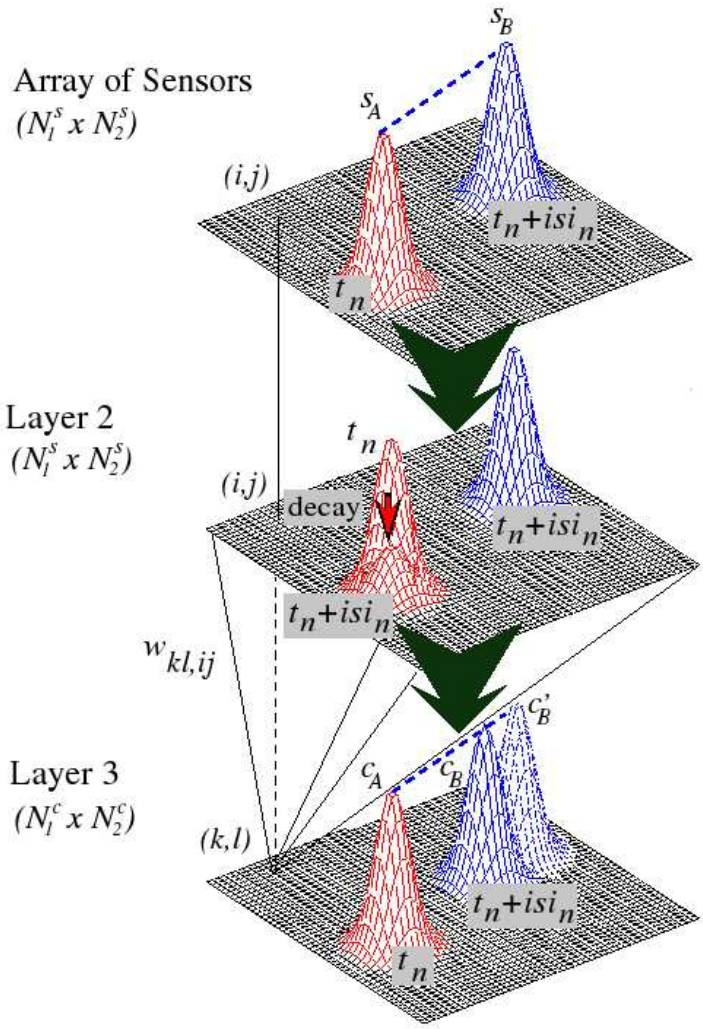


FIG. 1 – Architecture en trois couches du modèle TIS.

complexe et coûteux en calculs.

L'algorithme TOM est une simplification de l'algorithme TIS ; il se base sur la même idée de passage du temporel vers le spatial à l'aide d'une propagation de vague, mais les calculs se font de manière centralisée plutôt que distribuée, et de nombreux paramètres sont supprimés. Cette simplification rend l'algorithme TOM plus léger, ce qui contrebalance des défauts de TIS. En effet, l'algorithme TIS, contrairement à l'algorithme SOM, ne peut pas traiter des vecteurs stimuli de grandes dimensions.

L'algorithme TOM fournit un compromis entre les propriétés novatrices de TIS et l'efficacité de SOM. Comme nous le verrons plus tard, le compromis se fait au prix de la crédibilité biologique du modèle.

4 Description de l'algorithme

Présentons maintenant le déroulement de l'algorithme TOM de manière formelle.

4.1 Déroulement

L'algorithme TOM fonctionne avec les paramètres suivants, pour une carte et un corpus donnés :

- κ et σ_k, paramètres influençant respectivement la force initiale et la portée de l'interaction latérale,
- σ_0, l'intensité initiale du bruit,
- σ_f, l'intensité finale du bruit,
- n'_f, le pas de temps auquel le bruit se stabilise,
- v, la vitesse de propagation,
- α, le taux d'apprentissage,
- n_f, le nombre d'itérations de l'apprentissage.

La figure 2 schématise le déroulement d'une itération de l'algorithme sur une carte 1D.

Initialisation Les valeurs des vecteurs poids des neurones sont générées aléatoirement selon une distribution uniforme dans l'intervalle $[0, 1]$. Les valeurs des stimuli traités par le réseau sont elles aussi bornées sur le même intervalle.

Afin d'avoir une interaction temporelle à la première étape, on tire au hasard un stimulus et un vainqueur précédents.

Tirage du stimulus A chaque pas de temps le choix du stimulus dans le jeu de données se fait selon une distribution uniforme.

Exécution de l'étape n

1. On tire un stimulus s_n dans le jeu de données ; comme dans l'algorithme SOM décrit précédemment, on détermine le neurone vainqueur du *feed-forward* $k_{ff}(n) = argmax_k(w_k(n).s_n)$. La stimulation d'un neurone est

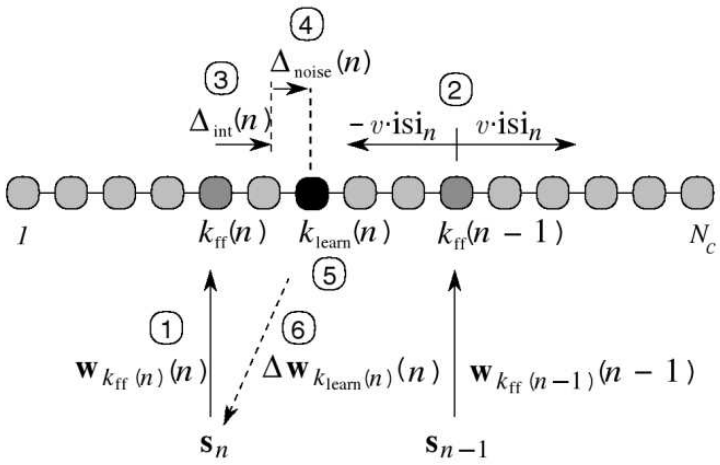


FIG. 2 – Schéma d'une itération de l'algorithme TOM sur une carte 1D

inversement proportionnelle à la distance euclidienne entre le stimulus s_n et $w_k(n)$, le vecteur poids du neurone k à l'instant n.

2. Le neurone $k_{ff}(n-1)$ activé par le stimulus s_{n-1} a entraîné la propagation d'une vague dans la carte. Cette vague se propage pendant un temps $isi_n = isi(s_{n-1}, s_n)$.
   $isi_n * v$ est la distance à laquelle la vague s'est propagée. Les coordonnées des deux "neurones de la vague" (coordonnées arrondies à l'entier le plus proche) sont calculées comme suit :

$$\tilde{k}_{\pm}(n) = k_{ff}(n-1) \pm v.isi_n \tag{1}$$

3. l'interaction entre le neurone de la vague le plus proche de $k_{ff}(n)$ provoque une déviation $\Delta_{int}(n)$ de $k_{ff}(n)$.
   Soit $\tilde{k}(n)$ le neurone de la vague $\tilde{k}_{\pm}(n)$ le plus proche de $k_{ff}(n)$;
   Soit $k = \tilde{k}(n) - k_{ff}(n)$ l'écart entre les coordonnées des deux neurones ;
   L'interaction entre le neurone vainqueur et la vague est exprimée par la fonction d'interaction suivante (fig. 3), où κ et σ_k sont des paramètres de la simulation :

$$f(k) = \kappa.tanh(\frac{k}{\kappa}).exp(-\frac{k^2}{2\sigma_k^2}) \tag{2}$$

   La fonction $f(k)$ permet de calculer $\Delta_{int}(n)$,

4. $\Delta_{noise}(n)$ est le bruit généré à l'étape n. Il est issu d'une distribution normale centrée en 0 et fonction du paramètre $\sigma_{noise}(n)$:

$$\Delta_{noise}(n) \sim \mathcal{N}(0, \sigma_{noise}(n)) \tag{3}$$

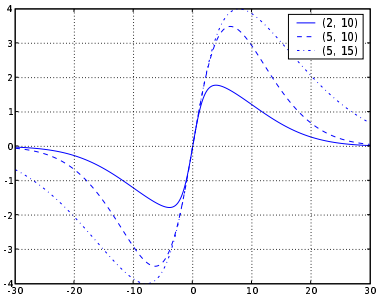


FIG. 3 – Fonction d'interaction latérale

La fonction $\sigma_{noise}(n)$ est une fonction décroissante avec le temps qui se stabilise arrivé à un nombre d'itérations donné, exprimée par les équations suivantes :

$$\sigma_{noise}(n) = \sigma_0.(\frac{\sigma_f}{\sigma_0})^{n/n'_f}, n <= n'_f \tag{4}$$

$$\sigma_{noise}(n) = \sigma_f, n'_f < n <= n_f \tag{5}$$

5. La position du neurone apprenant est déterminée en fonction du neurone vainqueur du feedforward et des deux déviations calculées :

$$k_{learn}(n) = round(k_{ff}(n) + \Delta_{int}(n) + \Delta_{noise}(n)) \tag{6}$$

6. Le neurone apprenant adapte son vecteur poids pour tendre vers le vecteur stimulus s_n, avec un taux d'apprentissage α :

$$\Delta w_{k_{learn}(n)}(n) = \alpha.(s_n - w_{k_{learn}(n)}(n)) \tag{7}$$

4.2 Topologie spatiale et topologie temporelle

L'information traitée par l'algorithme TOM possède une double topologie, spatiale et temporelle. Ces topologies permettent de situer les stimuli du jeu de données les uns par rapport aux autres en fonction de critères différents :

- la topologie spatiale en fonction des similarités entre les données composant les stimuli,
- La topologie temporelle en fonction de relations d'ordre liant les stimuli.

Les relations d'ordre composant la topologie temporelle dans l'algorithme TOM sont des intervalles temporels nommés "Intervalles Inter-Stimuli" (*Inter-Stimuli Intervals*, ou ISI). A toute paire de stimuli du jeu de données, on doit pouvoir associer une valeur d'ISI correspondant à l'écart temporel entre la présentation du premier et du deuxième stimuli. Lorsque ces deux stimuli

n'entretiennent pas de relation temporelle, on considère que l'ISI les séparant vaut ∞.

Séquence Les stimuli sont regroupés en séquences. A chaque stimulus, on associe une valeur d'ISI correspondant à l'écart temporel qui le sépare du stimulus précédent au sein de la séquence. Dans le cas où le jeu de données représente un phénomène particulier qu'on cherche à apprendre, on peut voir la séquence comme étant une instance du phénomène considéré.

Ensemble de séquences Pour représenter une information riche dans un jeu de données, une séquence n'est généralement pas suffisante. Un jeu de données est généralement constitué de plusieurs séquences. Les séquences ne contiennent pas nécessairement le même nombre de stimuli. On considère qu'il n'y a pas de relation temporelle entre les stimuli de deux séquences différentes.

4.3 Du temporel vers le spatial

L'algorithme TOM se fonde sur l'expression de l'information temporelle en distances sur la carte. Cette transposition se fait à l'aide d'une *propagation en vague*. Le neurone gagnant du *feedforward* en $t-1$, suite à la présentation d'un stimulus s_{t-1}, propage une vague dans la carte. La vague se propage jusqu'au prochain pas de temps, où sera présenté un stimulus s_t (fig. **??**).

La vitesse de propagation v est une constante définie au début de l'apprentissage. Elle permet de transposer $isi(t)$, l'écart temporel entre s_{t-1} et s_t, en une distance sur la carte : $d = isi(t) * v$. Cette information est plus à même d'influencer son organisation topographique.

La différenciation majeure entre SOM et TOM se situe après la détermination du vainqueur du feedforward. Alors que SOM provoque directement l'apprentissage du vainqueur, TOM calcule une déviation du vainqueur qu'on nomme *interaction latérale*. Celle-ci est provoquée par la vague propagée au pas de temps précédent.

L'interaction latérale dévie le neurone vainqueur afin de le rapprocher de la vague propagée. Suivant la position de la vague par rapport au neurone vainqueur et au neurone vainqueur précédent, la déviation aura deux effets possibles :

- éloigner le neurone vainqueur courant du neurone vainqueur précédent (ségrégation),
- ou le rapprocher (intégration).

Les paramètres d'interaction latérale κ et σ_k déterminent l'influence que la topologie temporelle a sur le processus d'apprentissage. Ces deux paramètres définissent la courbe de la fonction d'interaction (fig. 3). Cette fonction permet de calculer la déviation du neurone vainqueur en fonction de la distance qui le sépare de la vague propagée.

En traçant la courbe du rapport *déviation / distance* (fig. 4), on peut constater que la distance obtenue en sortie est toujours inférieure à la distance donnée en entrée. La déviation rapproche le neurone vainqueur de la vague, sans jamais

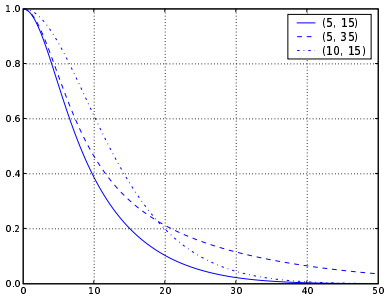


FIG. 4 – Rapport *déviation / distance* en fonction des paramètres κ et σ_k

l'atteindre. De plus, les paramètres κ et σ_k déterminent la vitesse de décroissance de l'influence : κ définit la force de l'interaction latérale sur les distances courtes, tandis que σ_k détermine la portée à laquelle l'interaction disparaît.

On précisera que dans ces calculs de déviation, on ne manipule pas des neurones mais des coordonnées de neurones. Lorsqu'il est nécessaire d'effectuer un apprentissage, on arrondit les coordonnées pour trouver dans la carte le neurone le plus proche.

4.4 Déviation due au bruit

En plus de l'interaction latérale, l'algorithme TOM possède une deuxième caractéristique qui le différencie de l'algorithme SOM. Alors que SOM applique un apprentissage à l'unisson centré sur le neurone vainqueur du feedforward (plusieurs neurones apprennent, avec des taux d'apprentissage décroissant en fonction de leur éloignement du neurone vainqueur), l'algorithme TOM, quant à lui, ne fait apprendre qu'un seul neurone par itération. Mais, pour unifier l'apprentissage, il laisse parfois un voisin du neurone vainqueur gagner. Ce choix se fait à l'aide d'un bruit issu d'une distribution normale, qui va provoquer une déviation du neurone apprenant, sur le même principe que l'interaction latérale. L'intensité du bruit est fonction d'un paramètre qui décroît avec le temps, pour permettre la convergence.

Wiemer avance que le remplacement de l'apprentissage à l'unisson, utilisé par SOM, par un apprentissage unique bruité offre de meilleurs résultats, en s'appuyant sur les travaux de [14]. Il en décrit l'effet ainsi :

> [...] the decrease of noise allows an annealing process to take place : first, topography is roughly determined with high noise level ; then, topography is successively refined by reducing the noise level [...] ; finally, topography is allowed to attain an equilibrium state by keeping the noise level adequately small [...]

L'écart type de la loi normale qui génère le bruit à chaque pas de temps décroît tout au long de la simulation (fig. 13). Les paramètres entrant en jeu

sont le bruit initial σ_0, le bruit final σ_f et le point de stabilisation du bruit n'_f. Wiemer décrit ces paramètres comme n'étant pas critiques pour l'apprentissage, pour peu qu'ils aient des valeurs crédibles :

- dans le modèle 1D, le bruit initial doit être environ égal au nombre de neurones de la carte ;
- le bruit final doit être suffisamment faible pour permettre à la carte de se stabiliser correctement - la valeur choisie est généralement 0.01 ;
- la stabilisation du bruit se fait à 90% de la simulation.

Troisième partie

Discussions et contributions

Nous allons maintenant étudier le comportement de l'algorithme TOM, dans des cas généraux ou limites. Ceci nous permettra de voir l'influence des paramètres de l'algorithme, de mettre en avant certaines limites, et de proposer des solutions ou des alternatives.

1 Jeux de données

Pour nos tests, nous utiliserons une série de jeux de données que nous présentons ici. Les résultats des tests sont présentés sous la forme de courbes, obtenues en faisant la moyenne de 5 exécutions. Les marges d'erreurs indiquées sont les écarts types.

1.1 Diagonale

Ce jeu de données (fig. 5) est composé d'une seule séquence de stimuli à une dimension. L'ISI entre deux stimuli consécutifs vaut 1. Chaque stimulus est représenté par un vecteur, et tous sont orthogonaux. Ce jeu de données permet de mettre la norme euclidienne en échec, afin d'étudier l'auto-organisation issue de la topologie temporelle.

1.2 Diagonale diffuse

La diagonale diffuse a le même format que la diagonale présentée précédemment. La seul différence réside dans le fait que les stimuli ne sont pas orthogonaux. Ici, l'organisation induite par la topologie spatiale est la même que celle induite par la topologie temporelle. (fig. 6

1.3 Topologies discordantes

Dans ce jeu de données composé d'une séquence, l'organisation issue de la topologie spatiale s'oppose à celle issue de la topologie temporelle (fig. 7

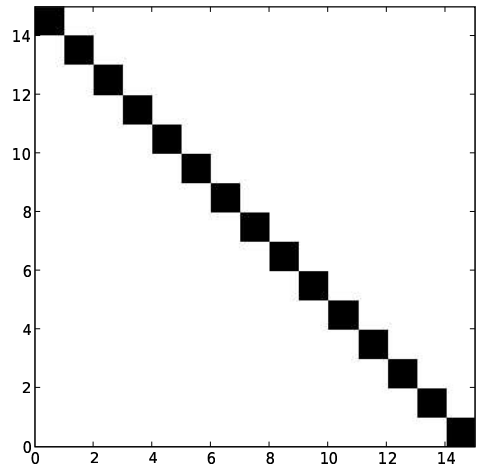


FIG. 5 – Diagonale : chaque ligne correspond à un stimulus. L'ordre des lignes forme la séquence.

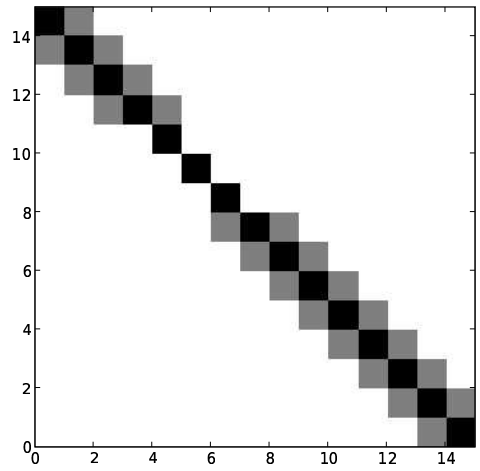


FIG. 6 – Diagonale diffuse : chaque ligne correspond à un stimulus. L'ordre des lignes forme la séquence.

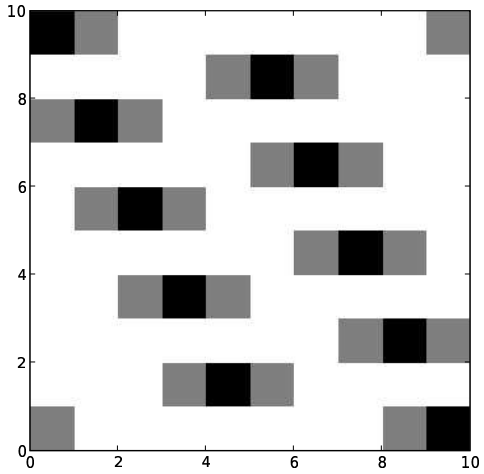


FIG. 7 – Topologies discordantes : chaque ligne correspond à un stimulus. L'ordre des lignes forme la séquence.

On qualifie ce cas de topologies discordantes. Dans la pratique, les topologies composant un jeu de données ne sont jamais totalement convergentes, ni même aussi opposées que le cas présenté ici.

1.4 Chiffres manuscrits

Afin de faire des essais d'apprentissage sur des données non triviales, nous utilisons un jeu de données composé d'un ensemble de dix séquences. Chaque séquence représente la suite des chiffres de 0 à 9. Chaque chiffre est représenté par une vignette de dimension 28x28 dans laquelle est stockée une version manuscrite (fig 8).

2 Analyse de l'auto-organisation

2.1 Organisation spatiale, organisation temporelle

Les données traitées par l'algorithme TOM possèdent une double topologie, à la fois spatiale et temporelle. L'auto-organisation menant à une topographie spatiale, l'information de la topologie temporelle est transformée afin d'être exprimable spatialement - c'est là le rôle de la propagation en vague, qui convertis les écarts temporels en distance spatiales. Notre auto-organisation est donc double.

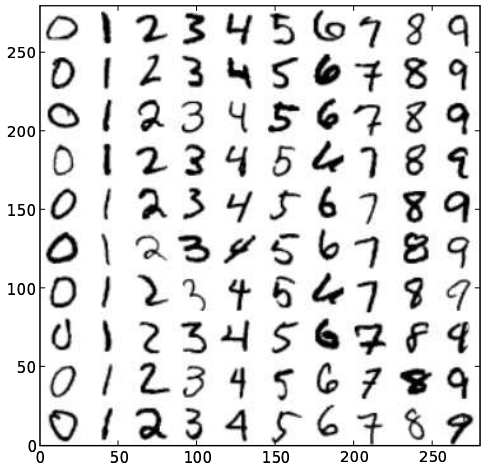
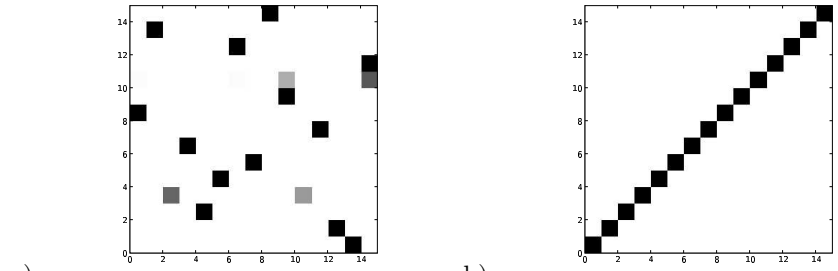


FIG. 8 – Chiffres manuscrits : chaque ligne représente une séquence de dix chiffres. l'ordre des vignettes au sein de chaque ligne définit l'ordre de la séquence. Les intervalles temporels entre deux stimuli voisins valent toujours 1.

Auto-organisation spatiale Spatialement, l'algorithme TOM a le même comportement que l'algorithme SOM : par l'usage d'une mesure donnée, on est capable d'estimer la similarité spatiale de deux stimuli. Cette mesure oriente l'apprentissage de telle sorte que la carte converge vers un état où des neurones voisins spatialement apprennent des stimuli similaires.

Auto-organisation temporelle La topologie temporelle influence l'auto-organisation spatiale en altérant le choix du neurone apprenant. Cette altération est fonction des paramètres de la conversion des écarts temporels du jeu de données vers des distances spatiales sur la carte.

Suppression de l'auto-organisation spatiale L'auto-organisation spatiale est tributaire de la mesure choisie. Si celle-ci n'est pas capable de quantifier le degré de similitude entre les stimuli, il n'y aura pas auto-organisation. Dans notre travail, nous avons utilisé une norme euclidienne pour exprimer cette distance. Pour mettre la norme euclidienne en échec, il suffit que tous les vecteurs stimuli soient orthogonaux dans l'espace d'entrée. Alors, les stimuli seront appris par la carte, mais leur disposition sera aléatoire. En choisissant un jeu de données qui mette en échec la norme euclidienne, on supprime l'auto-organisation spatiale. La topographie obtenue alors n'est issue que de l'influence de la topologie temporelle.



a) b)

FIG. 9 – Apprentissage du jeu de données “diagonale” sur une carte de 15x1. Chaque ligne correspond au vecteur poids d'un neurone. (a) apprentissage purement spatial, équivalant à un SOM classique. (b) apprentissage spatio-temporel - la topographie de l'apprentissage respecte l'information temporelle issue du jeu de données.

Prenons une carte de neurones à une dimension de longueur 15, et faisons lui apprendre un jeu de données diagonal (fig. 5) de longueur 15. Dans le cas d'un apprentissage purement spatial, la carte convergera vers un état où chaque neurone a appris un stimulus du jeu de données. La disposition de cet apprentissage est fonction des poids initiaux et de la loi de probabilité utilisée pour la présentation des stimuli (fig. 9a). En revanche, si il y a influence du temporel temporel par le biais de l'interaction latérale, on constate que l'organisation de la carte reflète la séquence du jeu de données diagonal (fig. 9b).

2.2 Concordance et discordance

L'exemple précédent nous montre que la topographie de la carte peut être influencée par la topologie temporelle. On a donc bien une double auto-organisation. Mais nous sommes dans un cas limite où il n'y a pas d'auto-organisation spatiale. Quand celle-ci existe, on peut se demander comment les deux auto-organisations vont cohabiter au sein de la carte. Pour présenter ce processus, on présentera deux cas extrêmes.

Concordance Nous avons vu précédemment, par l'usage du jeu de données diagonal (fig. 5), que l'information temporelle a pour effet d'influencer la disposition de l'apprentissage spatial au sein de la carte. Il existe des situations où les topographies visées par la topologie spatiale et la topologie temporelle sont similaires, ou au moins compatibles. Les deux auto-organisations convergent alors pour mener à une topographie spatiale donnée.

Le jeu de données de la diagonale diffuse (fig. 6) en est un exemple. L'apprentissage de ce jeu de données par une carte à une dimension de même longueur que la séquence mène à une topographie à l'image de la diagonale diffuse (fig. **??**). Sur un passage de la séquence, l'auto-organisation spatiale n'est plus pos-

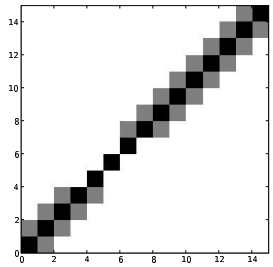


FIG. 10 – Apprentissage du jeu de données “diagonale diffuse” sur une carte de 15x1. Chaque ligne correspond au vecteur poids d'un neurone. La topographie finale est à l'image du jeu de données d'apprentissage.

sible car l'information est appauvrie. On voit sur le résultat de l'apprentissage spatio-temporel que la section appauvrie de la séquence a bien convergé grâce au concours de l'information temporelle.

Discordance La plupart du temps, les organisations issues des topologies spatiales et temporelles ne sont pas entièrement compatibles. Le jeu de données “topologies discordantes” (fig. 7) est un cas limite où elles sont totalement incompatibles : en suivant la topologie spatiale uniquement, la carte converge vers une topographie en diagonale diffuse ; en revanche, en suivant la topologie temporelle uniquement, la carte converge vers une topographie à l'image de la séquence.

Cas général Le résultat de cette double auto-organisation est un consensus entre les deux topologies. Le rapport de force entre le spatial et le temporel illustré de la cas de la discordance se fait en fonction des paramètres choisis pour l'apprentissage.

2.3 Mesure de l'organisation

Les discussions que nous entreprenons dans cette partie nécessitent de quantifier l'auto-organisation d'une carte. Si, sur des exemples triviaux, on peut s'en faire une idée en observant l'état final de la carte c'est rarement le cas dans les cas plus complexes. Nous avons donc besoin d'une mesure numérique quantifiant la qualité de l'auto-organisation, afin de comparer différentes situations.

Le cadre de l'auto-organisation pose une première difficulté ; nous n'avons pas d'état final cible auquel comparer le résultat de l'apprentissage. La deuxième difficulté tient au fait que la topographie de la carte est issue d'une double auto-organisation, à la fois spatiale et temporelle. On devrait donc avoir une mesure pour chaque convergence.

Il n'est pas aisé de juger la qualité de l'organisation d'une carte. Pour nos mesures, nous avons choisi d'opérer en effectuant des projections de l'ensemble des données d'apprentissage sur la carte.

2.3.1 Organisation spatiale

Dans un premier temps, mettons en place une mesure qualité l'organisation spatiale de la carte. Notre procédé se base sur la considération suivante : l'apprentissage dans l'algorithme TOM est du type *winner-take-all*. A chaque stimulus présenté, il n'y a qu'un seul neurone vainqueur. Ceci a pour conséquence de rassembler les stimuli similaires à un seul endroit sur la carte. A chaque fois qu'un stimulus est présenté, il renforce une zone de la carte, qui augmentera ses chances de remporter à nouveau à la prochaine présentation du stimulus. Itérativement, il devient de moins en moins probable que deux neurones éloignés sur la carte soit très similaires.

Notre mesure consiste à utiliser le feedforward pour projeter un stimulus s du jeu d'apprentissage sur la carte. En retour, on obtient un classement $proj_s$ des neurones de la carte, du plus proche de s au plus éloigné.

De $proj_s$, on ne garde que les meilleurs neurones pour former $proj'_s$. On prend pour référence le score du meilleur neurone noté win_s, et on ne garde que les neurones dont le score est inférieur à $win_s * \alpha, \alpha >= 1$.

Chaque neurone possède des coordonnées sur la carte. L'écart type de ces coordonnées nous donne une mesure de la dispersion des neurones de la projection. En plus de l'écart type, on calcule le barycentre de la projection, que l'on considère comme étant le projeté du stimulus sur la carte.

Si s a mal été appris par la carte, alors win_s sera élevé. En conséquence, sa projection sera grande et disparate, donnant un écart type élevé.

En revanche, si s a bien été appris par la carte, alors win_s est faible. Sa projection sera donc plus restreinte et groupée, donnant un écart type faible.

Notre mesure globale consiste à calculer l'écart type des projetés de tous les stimuli du jeu d'apprentissage, et d'en faire une moyenne. Un résultat faible indique une bonne organisation spatiale.

2.3.2 Topographie temporelle

Comme nous l'avons montré précédemment, l'organisation de la carte est influencée par la topologie temporelle par le biais de l'interaction latérale, de telle sorte que les projetés des stimuli du jeu de données soient positionnés selon une disposition qui satisfasse au mieux la contrainte exercée par la topologie temporelle. On peut donc considérer que l'apprentissage de la topologie temporelle est optimal lorsque les projections des stimuli satisfont au mieux les contraintes issues de la topologie temporelle.

Les calculs de projection de la partie précédente nous ont permis d'obtenir des points sur la carte associés aux stimuli du jeu de données d'apprentissage. De plus, le jeu de données nous offre comme information les intervalles temporels séparant les stimuli. Nous avons vu que la propagation en vague permet de

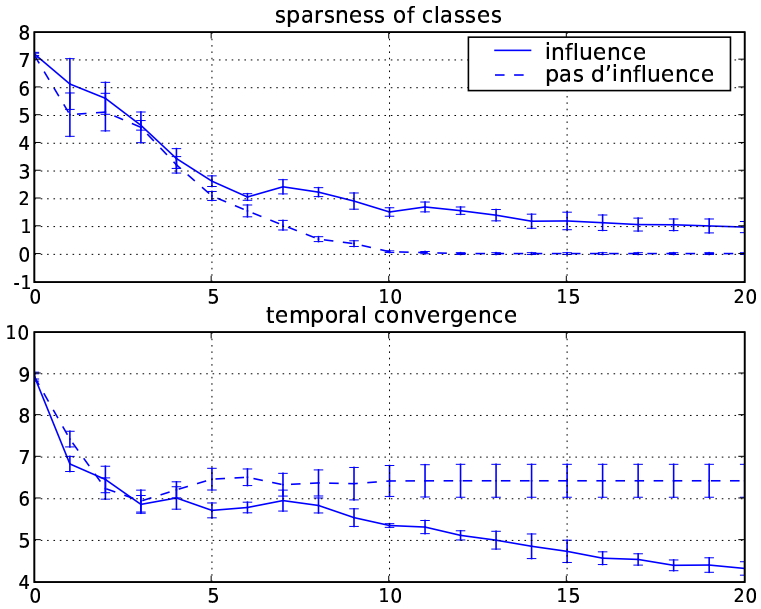


FIG. 11 – Apprentissage sur le jeu de données des caractères manuscrits. La fenêtre du haut présente l'évolution de l'éparpillement des projections. La fenêtre du bas la mesure de convergence temporelle. Les deux courbes montrent respectivement un apprentissage avec et sans interaction latérale. L'interaction latérale tend à déformer les projections (et donc à augmenter leurs écarts type), comme l'indique la mesure de dispersion ; la mesure de convergence temporelle montre que l'interaction latérale a un effet certain sur le positionnement des projections.

convertir la distance temporelle en distance spatiale selon un facteur v, la vitesse de propagation de la vague.

Prenons maintenant deux stimuli s_1 et s_2 d'une séquence du jeu de données d'apprentissage. la distance temporelle $t = isi(s_1, s_2) * v$. s_1 et s_2 ont deux projetés p_{s1} et p_{s2} sur la carte calculés par la mesure de convergence spatiale. La distance spatiale $s = |p_{s1}p_{s2}|$. Si la topographie de la carte respecte la topologie temporelle, alors $t \approx s$.

La mesure de convergence temporelle consiste donc à calculer, pour chaque paire de stimuli, $|d - s|$, et à faire la moyenne de toutes ces valeurs.

Dans le cas où le jeu de données comprend plusieurs séquences, on ne considère naturellement que les paires de stimuli issues de la même séquence - les autres n'entretenant pas de relations temporelles.

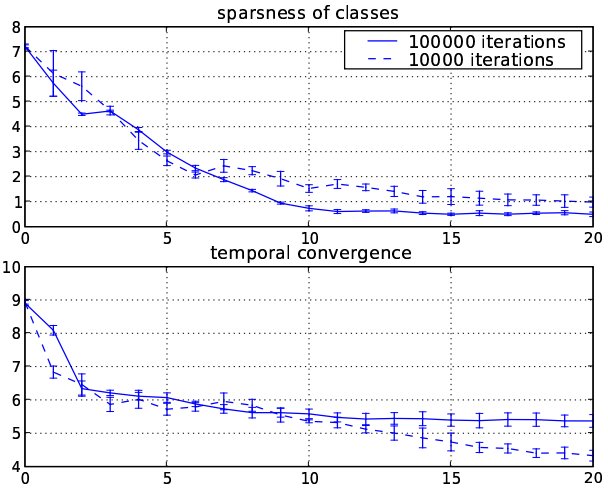


FIG. 12 – Comparaison entre une exécution sur 100000 itérations avec un taux d'apprentissage de 0.01 et une exécution de 10000 itérations avec un taux de 0.1. Les deux courbes sont reportées sur mises à la même échelle pour la comparaison.

2.4 Influence des paramètres sur l'auto-organisation

2.4.1 Taux d'apprentissage

Le taux d'apprentissage détermine à quel point un neurone adapte ses poids lorsqu'on lui présente un stimulus. Ce paramètre est très lié avec le nombre d'itérations de l'apprentissage, ainsi que le nombre de neurones de la carte. Wiemer privilégie un apprentissage lent dans son usage de l'algorithme TOM - ses exemples jouets tournent sur 100000 itérations, avec un taux d'apprentissage de 0.01. On constate sur la figure 12 que l'apprentissage long améliore la convergence spatiale, mais dégrade la convergence temporelle.

2.4.2 Vitesse de propagation

La vitesse de propagation des vagues au sein de la carte est un paramètre très influent pour l'organisation temporelle de la carte : elle détermine l'échelle de la transposition de l'écart temporel en distance spatiale. Nous avons vu que la vague partie en $t-1$ provoque une déviation du neurone apprenant en t. Suivant la position de la vague par rapport au neurone apprenant, cette déviation est une intégration ou une ségrégation. La ségrégation ayant lieu lorsque la vague a dépassé la position du neurone apprenant, plus la vitesse de propagation est grande, plus la proportion de ségrégation augmentera.

Les ségrégations provoquant un étalement de l'apprentissage au sein de la carte, la vitesse de propagation permet de contrôler la diffusion de l'apprentis-

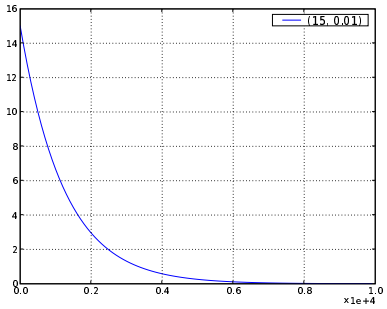


FIG. 13 – Courbe de décroissance du paramètre $sigma_{noise}$ au cours de la simulation

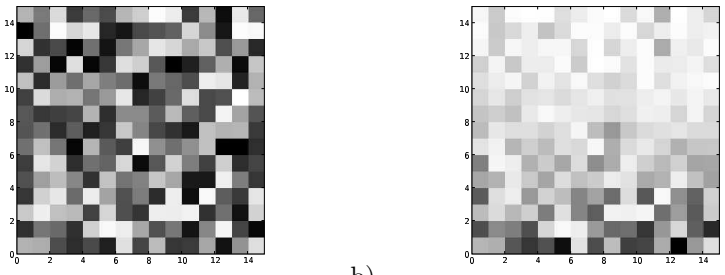


FIG. 14 – État d'une carte de 15x1 à différents moments de son apprentissage. (a) état initial de la carte. (b) état à 5% de la simulation.

sage sur la carte.

2.4.3 Bruit

Le bruit fort au début de l'apprentissage sert d'étape de préparation. L'apprentissage étant très bruité, les neurones sont amenés à converger vers un état qui sera une moyenne de tous les stimuli du jeu de données. Faire correspondre le bruit initial et la taille de la carte permet d'adapter la superficie de la zone normalisée aux dimensions de la carte. En effet, si le bruit initial est plus faible, cette normalisation ne se produira que sur une partie réduite de la carte (fig. 14). La décroissance du paramètre de bruit permet un affinement progressif de l'organisation de la carte.

Cependant, en plus d'être biologiquement peu probable, nous constatons expérimentalement que le rôle uniformisant du bruit initial fort n'est pas toujours parfaitement rempli.

A l'initialisation, tous les neurones ont des vecteurs poids aléatoires, les rendant généralement tous très éloignés des stimuli du jeu d'apprentissage. Dans

cette situation, lorsqu'un neurone apprend d'un stimulus quelconque du jeu d'apprentissage, il y a de fortes chances pour qu'il se rapproche de tous les stimuli du jeu de données.

Lorsque cet effet est combiné avec un bruit initial fort, on se retrouve dans une situation où quel que soit le neurone vainqueur du feedforward, il y a de fortes chances pour que le bruit le déporte jusqu'à atteindre un bord de la carte. C'est alors ce neurone qui apprend et se rapproche de tout le jeu de données, augmentant ses chances de remporter le prochain feedforward. Lorsqu'un neurone de bord de carte remporte le feedforward, il est sujet à la déviation due au bruit. Mais sa position sur le bord de la carte le rend beaucoup moins influençable que les autres neurones. Par conséquent, il est plus probable que ce dernier soit à nouveau le neurone apprenant, renforçant encore sa position.

A cause de cela, la phase d'uniformisation de la carte ne fonctionne pas toujours bien. Généralement, on voit plutôt cette uniformisation se propager depuis un bord de la carte vers le reste. Une propagation similaire est obtenue avec un bruit plus faible et une interaction latérale forte.

2.4.4 Rapport entre les paramètres

La taille de la carte nécessaire à l'apprentissage optimal d'un jeu de données n'est généralement pas une information possédée avant l'apprentissage. Dans l'algorithme TOM, lorsqu'une carte trop grande est initialisée avec un bruit fort, on constate un sur-apprentissage des extrémités de séquences. En revanche, lorsque le bruit est plus faible, l'initialisation de la carte se fait progressivement et l'apprentissage a tendance à se limiter à une sous-partie de la carte.

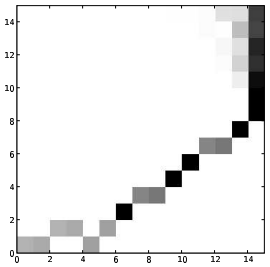
Nous avons vu précédemment que la vitesse de propagation nous permet de contrôler la diffusion de l'apprentissage. Par extension, l'écart temporel le plus grand du jeu de données nous donne un indicateur de la diffusion maximale de l'apprentissage. Ceci peut nous donner une base pour choisir les paramètres :

- soit isi_{max} l'ISI maximum du jeu de données,
- soit v la vitesse de propagation de la vague,
- soit l la longueur de la carte,
- utiliser l'égalité $isi_{max} * v = l$ nous permettra d'adapter la vitesse de propagation à la taille de la carte, ou inversement.

De plus, si on souhaite que l'apprentissage se réduise à une sous-partie de la carte, il suffit de choisir l comme étant la dimension de l'espace d'apprentissage souhaité, et d'adapter la vitesse de propagation en conséquence. Dans une telle situation, il parait plus judicieux d'indexer le bruit initial sur la valeur l plutôt que la taille de la carte.

Les extrémités de séquences forment les couples de stimuli les plus éloignés temporellement ; l'organisation temporelle tend donc à les positionner aux extrémités de la zone d'apprentissage. Le bruit diffusant l'apprentissage, les extrémités de séquences tendent à être apprises au delà des limites de l'organisation temporelle. Mais lorsque le bruit décroît et que l'auto-organisation s'affine, la diffusion s'affaiblit, laissant le contour de la zone d'apprentissage dans un état d'apprentissage partiel.

a)



b)

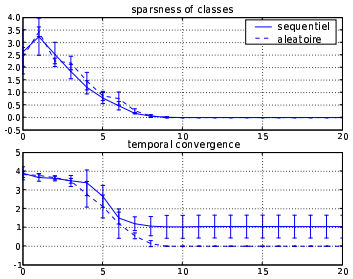


FIG. 15 – Apprentissage du jeu de données “diagonale” présenté séquentiellement, par une carte de 15x1. (a) état final d'une carte. On constate la présence d'une diagonale dégénérée. (b) comparaison de l'apprentissage séquentiel avec un apprentissage aléatoire classique. Perte de qualité du point de vue de l'organisation temporelle.

3 Modifications de l'algorithme

3.1 Ordre de présentation des stimuli

L'algorithme SOM, qui est la base sur laquelle les modèles TIS et TOM sont construits, utilise une loi de probabilité pour choisir le stimulus à présenter lors de chaque pas de temps de l'apprentissage. Ce tirage stochastique est aussi utilisé par les modèles TIS et TOM. Il nous paraît étonnant que, pour une méthode tirant son inspiration du traitement de flux sensoriels, on fasse usage d'un tirage aléatoire pour recalculer ensuite les distances temporelles en fonction de la séquence ainsi générée. La crédibilité biologique du modèle en est réduite, et cette approche ferme des possibilités telles que l'apprentissage permanent sur des données temps réel.

C'est pourquoi, en vue de s'en passer, nous étudions l'importance du tirage stochastique pour la convergence de la carte. Pour cela, nous le remplaçons par un tirage séquentiel des stimuli, suivant leur ordre d'apparition au sein de la séquence. Ceci correspond mieux au traitement de données arrivant en continu. Dans un premier temps, nous nous contenterons d'un jeu de données ne comprenant qu'une seule séquence simple - nous prendrons le jeu de données diagonal (fig. 5).

Les résultats de l'apprentissage (fig. 15) montrent graphiquement une perte de qualité sur le plan temporel, confirmée par notre mesure. Localement, la carte obtenue semble avoir convergé correctement. Mais globalement, on se rend compte de la présence d'artéfacts.

Pour comprendre ce phénomène, observons les interactions latérales qui opèrent. Soient deux pas de temps t et $t+1$: le stimulus s_t est la source de l'interaction latérale qui va interagir avec le stimulus s_{t+1}. Avec un tirage stochastique, n'importe quel stimulus peut sortir en t et $t+1$. Chaque stimulus est donc susceptible d'influencer tous les autres. Un grand nombre de pas

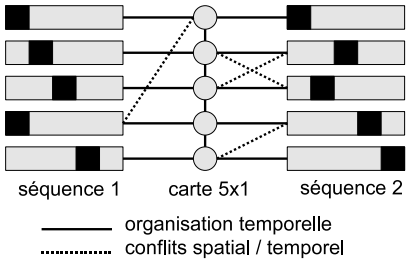


FIG. 16 – Un apprentissage sur ces deux séquences simples entraîne des problèmes car l'additivité des ISI, qui permet de disposer l'apprentissage de manière linéaire, ne fonctionne plus.

de temps et un tirage uniforme entraîneront une uniformisation des influences temporelles entre les stimuli. Pour n stimuli, le nombre d'interactions possibles s'élève à n^2.

En revanche, dans le cas d'un tirage séquentiel, on appauvrit grandement l'interaction latérale. L'ordre étant toujours le même, le nombre d'interactions possibles descend à n. Il est important d'ajouter que les interactions seront toujours locales ; les stimuli étant présentés séquentiellement, les valeurs d'ISI seront toujours minimales (à l'exception du bouclage de la séquence). La disposition des projections sera donc influencée temporellement à un niveau local, mais aucunement à un niveau global.

Sur des exemples jouets tels que ceux utilisés précédemment, l'effet n'est pas très important, pour plusieurs raisons :

- chaque stimulus peut être appris par un seul neurone,
- il n'y a qu'une seule séquence accompagnée d'ISI, et le calcul des ISI respecte la loi d'additivité.

On en déduit que si les projections sont positionnées correctement deux à deux, alors toute la séquence est disposée correctement. Une petite erreur engendrée par le bruit au niveau d'un positionnement provoquera un décalage dans la séquence.

En revanche, sur un jeu données moins trivial, le rapport entre le spatial et le temporel est plus complexe. Ainsi, des stimuli similaires spatialement (supposés avoir des projections proches sur la carte) peuvent entretenir des relations temporelles différentes suivant la séquence considérée.

La propriété d'additivité des ISI au sein d'une séquence permettait de la transposer spatialement au sein de la carte ; mais avec une séquence contenant des doublons, ou plusieurs séquences, cette transposition n'est plus possible. Alors, la convergence temporelle locale ne garantit plus une convergence globale correcte (fig. 16).

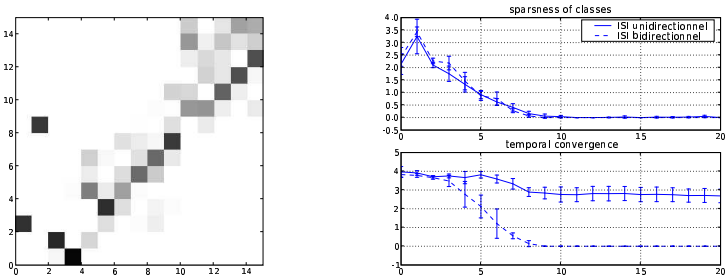


FIG. 17 – Apprentissage du jeu de données "diagonale" avec un ISI unidirectionnel, par une carte de 15x1. (a) état final d'une carte. (b) comparaison de l'apprentissage unidirectionnel avec un apprentissage aléatoire classique. Perte de qualité du point de vue de l'organisation temporelle.

3.2 Influence temporelle rétroactive

Dans l'algorithme TOM, le calcul de l'ISI séparant deux stimuli ne prend pas en considération quel stimulus s'est produit en premier. L'ISI est un écart temporel qui ne prend pas en considération le sens du temps.

Pour a et b deux stimuli appartenant à la séquence s, tels que a est antécédent à b dans s, $ISI(b,a) = ISI(a,b)$. Si, au pas de temps $t-1$, $s_{t-1} = b$, et au pas de temps t, $s_t = a$, ceci revient à dire que b a une influence temporelle rétroactive sur a. Avec un tirage stochastique uniforme, cette situation représente 50% des cas d'interaction temporelle. Ceci peut être assimilé à un apprentissage à rebours, qui est considéré comme biologiquement peu probable [15]. Nous qualifions ce calcul de l'ISI de bidirectionnel.

Il est possible de modifier l'algorithme pour interdire ces influences temporelles radioactives. Pour cela, considérons que pour a et b deux stimuli appartenant à la séquence s, tels que a est antécédent à b dans s, $ISI(b,a) = \infty$, ce qui signifie que b n'a pas d'influence temporelle sur a. Nous qualifions ce calcul de l'ISI d'unidirectionnel. La figure 17 montre le résultat de ce type d'apprentissage sur l'exemple jouet de la diagonale.

On constate que la qualité de la convergence décroît au fur et à mesure qu'on se rapproche du début de la séquence. Bloquer l'influence temporelle rétroactive appauvrit l'influence de la topologie temporelle d'une manière particulière : chaque stimulus ne peut influencer que les stimuli qui le suivent au sein de la séquence. Ainsi, le placement du premier stimulus est entièrement issu de la topologie spatiale, car aucun des stimuli de la séquence n'a d'influence temporelle sur lui. Plus on avance dans la séquence, plus les stimuli sont influençables temporellement ; c'est pourquoi la fin de la séquence converge bien.

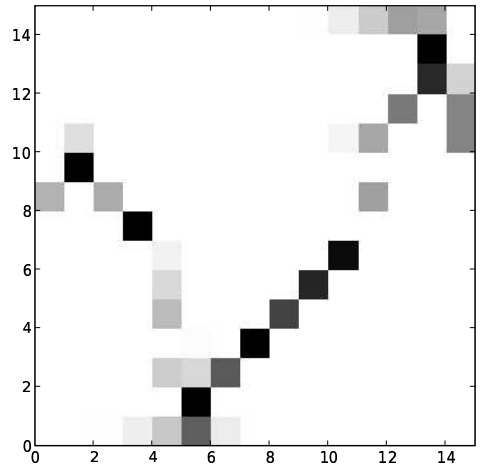


FIG. 18 – Apprentissage du jeu de données diagonal sur une carte de 15x1, avec présentation séquentielle et calcul unidirectionnel de l'ISI.

3.3 Homogénéité de l'influence temporelle

Les problèmes de convergence qui apparaissent lorsqu'on supprime le tirage aléatoire ou l'influence temporelle rétroactive mettent en avant la nécessité d'une influence temporelle riche et homogène, passant par des interactions latérales entre tous les stimuli du jeu d'apprentissage. Moins un stimulus est influencé temporellement, plus le positionnement de sa projection sur la carte se fait uniquement en fonction de l'organisation spatiale - on qualifiera cette zone de statique.

Une zone statique, qui reflète des stimuli peu ou pas influencés temporellement, ne subira que très peu les reconfigurations dues à l'influence de la topologie temporelle. En revanche, les stimuli se projetant dans cette zone auront une influence temporelle sur les autre stimuli, forçant une organisation de la carte en fonction de la zone statique.

Dans le cadre d'une carte aux dimensions limitées comme celles que nous utilisons ici, une zone statique peut poser de gros problèmes si elle entre en conflit avec un bord de la carte. Cet effet est particulièrement flagrant avec un apprentissage séquentiel et unidirectionnel sur une carte ne laissant pas d'espace à l'apprentissage (fig. 18). Le départ de la séquence n'étant pas appris à une extrémité de la carte, le reste de la séquence se positionne par rapport à ce point et se brise sur le bord de la carte, empêchant une convergence correcte. Si la projection du stimulus de début de séquence n'avait pas été statique, l'influence de la topologie temporelle l'aurait déporté vers le bord de la carte jusqu'à ce que la séquence ait suffisamment de place pour s'étendre correctement.

Biologiquement, il parait logique que l'apprentissage d'un stimulus situé en début de séquence ne soit pas influencé temporellement par des stimuli qui n'apparaîtront que plus tard. Dans le cadre d'un apprentissage distribué et à grande échelle, le début de séquence peut permettre de sélectionner et préactiver une zone particulière, au sein de laquelle la suite de l'apprentissage s'effectuera.

3.4 Tirage sur plusieurs séquences

Problème : Tirage au sein de plusieurs séquences L'algorithme TOM utilise une loi de probabilité pour choisir un stimulus à chaque pas de temps. Si le tirage se fait au sein de tous les stimuli, séquences confondues, il arrivera nécessairement que deux stimuli consécutifs n'appartiennent pas à la même séquence.

Dans un tel cas, l'influence temporelle sera nulle car l'ISI séparant les deux stimuli vaudra ∞. Plus généralement, la probabilité que deux stimuli consécutifs appartiennent à la même séquence est inversement proportionnelle au nombre de séquences qui composent le jeu de données. Ceci provoque un effondrement de l'influence temporelle dans le processus d'apprentissage qu'il est nécessaire d'empêcher.

Solution : apprentissage parallèle Afin de résoudre ce problème, nous avons choisis d'effectuer un apprentissage parallèle de toutes les séquences du jeu de données.

Soit d un jeu de données constitué de 3 séquences s_0, s_1, s_2.

- au pas de temps t_0, on sélectionne et on apprend avec un stimulus $stim_0$ tiré dans la séquence s_0
- au pas de temps t_1, avec un stimulus $stim_1$ tiré dans la séquence s_1
- au pas de temps t_2, avec un stimulus $stim_2$ dans la séquence s_2
- au pas de temps t_3, $stim_3$ est à nouveau tiré dans s_0, et l'apprentissage subis l'influence de la vague provoquée par le stimulus $stim_0$, qui se trouve être le dernier stimulus tiré au sein de la séquence s_0.
- et ainsi de suite

Par ce procédé, on évite l'effondrement de l'effondrement de l'interaction latérale. Un autre procédé consiste à apprendre les séquences les unes après les autres, mais ceci risquerait de provoquer des problèmes tels que l'oubli des séquences apprises précédemment, et l'inégalité de l'apprentissage des séquence dû à la décroissance du paramètre de bruit.

3.5 Propagation en vagues et dimensions de la carte

La distance à laquelle une vague se propage est fonction de la constante v, vitesse de propagation, et de isi_n l'intervalle inter-stimuli séparant s_n et s_{n-1}. Dans l'algorithme TOM tel qu'il est présenté par Wiemer, on choisit la source de l'interaction latérale $\tilde{k}(n)$ en fonction de la position de la vague au sein de la carte : les deux neurones les plus proches de la vague sont sélectionnés comme étant "neurones de la vague", et le neurone de la vague le plus proche de $k_{ff}(n)$,

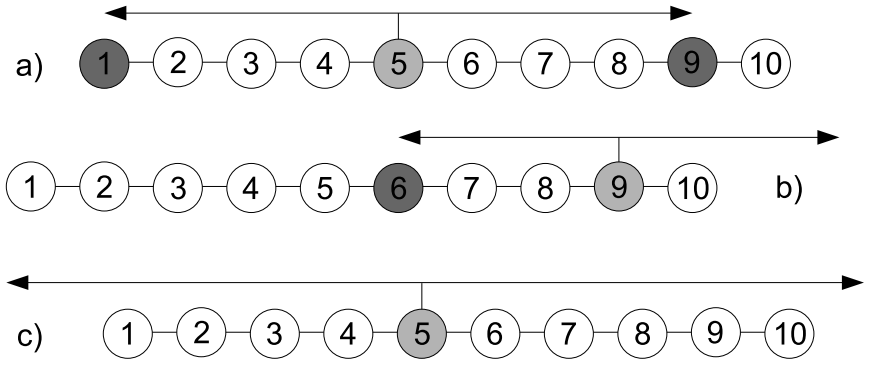


FIG. 19 – On distingue trois cas de positionnement de la vague sur la carte 1D : (a) la vague s'étend dans les limites de la carte et recouvre donc deux neurones ; (b) la vague sort partiellement de la carte et ne recouvre plus qu'un neurone ; (c) la vague sort totalement de la carte et ne recouvre pas de neurones.

vainqueur courant du feedforward, est la source de l'interaction latérale, $\tilde{k}_{\pm}(n)$ (fig. 19a).

3.5.1 Inversement du sens de l'interaction latérale

En revanche, si isi_n est plus grand, ou que la source de la vague est décalée, on peut se trouver dans une situation où il n'y a qu'un seul neurone de la vague. En toute logique, il devrait alors être le neurone de la vague le plus proche de $k_{ff}(n)$, et donc être pris comme $\tilde{k}(n)$. Mais la figure 19b nous montre que si la carte avait été plus longue et qu'un deuxième neurone de la vague avait existé, alors celui-ci aurait été choisi comme $\tilde{k}(n)$, et il en aurait résulté une interaction latérale dans le sens opposé. Cette considération faite, il apparaît peu judicieux de considérer la vague d'une manière qui inverserait son rôle.

3.5.2 Préservation d'intensité de la déviation latérale

Un troisième cas de figure est possible : l'isi_n est tellement élevé que la vague sort de la carte par les deux extrémités (fig. 19c). Alors, $\tilde{k}_{\pm}(n) = \emptyset$. Dans ce cas, doit-il y avoir une influence latérale ? Et si oui quelle source prendre ?

Une première solution à ces problèmes pourrait consister à appliquer une limite à la propagation de la vague, de manière à ce qu'elle butte sur les bords de la carte. Ainsi, sur la figure 19b, les neurones $\tilde{k}_{\pm}(n)$ sont les numéros 6 et 10. De même, sur la figure 19c, les neurones $\tilde{k}_{\pm}(n)$ sont les numéros 1 et 10.

Mais les distorsions provoquées par cette méthode pourraient causer des problèmes. En effet, quand une vague sort de la carte, quelle que soit la distance à laquelle la vague est supposée aller, l'intensité de l'interaction latérale qu'elle

génère ne décroît pas en dessous de la limite posée par la distance au bord de la carte.

3.5.3 Déviations fortes

Dans l'algorithme TOM, la distribution normale générant le bruit part avec un écart type fort, qui décroît tout au long de l'apprentissage. Il n'est pas rare d'obtenir en début de simulation des valeurs de bruit qui provoque une déviation des coordonnées du neurone apprenant $k_{learn}(n)$ dépassant les limites de la carte. Dans une telle situation, comment choisir le neurone apprenant ? Sur le cas 1D, la solution semble être de limiter la déviation du bruit aux limites de la carte. Un bruit initial fort aura alors pour conséquence de forcer l'apprentissage des neurones aux extrémités.

3.5.4 Solution : coordonnées théoriques

Pour résoudre ces problèmes, nous proposons la méthode suivante :

- pour résoudre le problème de la vague, plutôt que de prendre un "neurone de la vague" comme source de l'interaction latérale, nous prenons le point le plus proche de la vague elle-même, et ce sans considération des limites de la carte. Ainsi, $\tilde{k}(n)$ peut se trouver en dehors de la carte. On calcule normalement la déviation de $k_{ff}(n)$ due à l'interaction latérale. Cette technique évite l'inversion de l'interaction (fig. 19b), de même que la préservation de l'intensité (fig. 19c). On obtient alors un premier vecteur de déviation de $k_{ff}(n)$.
- Au vecteur ainsi calculé, on ajoute la déviation due au bruit.
- Le vecteur de déviation total ainsi calculé est appliqué depuis les coordonnées de $k_{ff}(n)$, nous donnant les coordonnées théoriques de $k_{learn}(n)$.

Il suffit d'ajouter le vecteur de bruit aux coordonnées du neurone apprenant pour obtenir son emplacement final. Mais, étant donnée la force du bruit, il se peut que ces coordonnées soient en dehors de la carte. Dans ce cas, $k_{learn}(n)$ est le neurone le plus proche des coordonnées théoriques, situé sur la droite entre $k_{ff}(n)$ et $k_{learn}(n)$ théorique. Dans le cas d'une carte 1D, il s'agit simplement du neurone à l'extrémité la plus proche de la carte.

3.6 Fenêtre temporelle de taille supérieure à un

A un instant t, l'apprentissage est influencé par la vague émise à l'instant $t-1$. On peut aisément modifier l'algorithme pour prendre en compte plusieurs vagues au lieu d'une seule ; ainsi, on définit une constante n, et l'apprentissage à l'instant t est influencé par les vagues issues de $t-1, ..., t-n$. Cet *apprentissage d'ordre n* peut être vu comme une surclasse de l'algorithme TOM : pour $n = 0$, il s'agit d'un SOM classique avec l'apprentissage à l'unisson remplacé par un apprentissage bruité ; pour $n = 1$, il s'agit d'un TOM classique.

Observons le comportement de l'apprentissage d'ordre 2. La figure 20 nous montre qu'il donne en moyenne de moins bons résultats, tant spatialement que

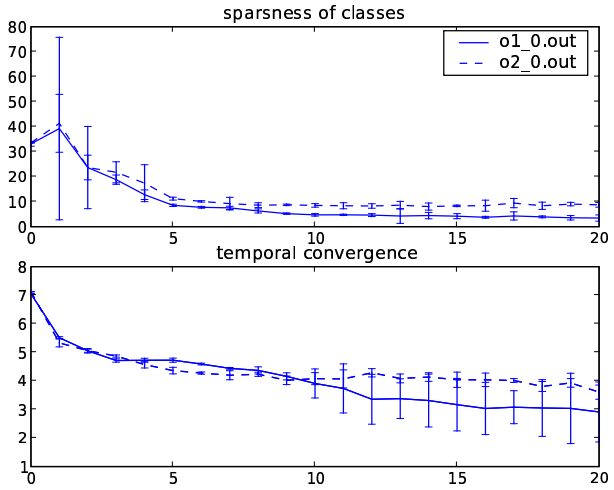


FIG. 20 – Comparaison entre l'apprentissage d'ordre 1 et d'ordre 2, sur le jeu de données "chiffres manuscrits". La carte fait 20x1.

temporellement. En revanche, il est intéressant de constater que l'écart type de la convergence temporelle est bien plus faible à l'ordre 2.

4 Extension aux cartes à deux dimensions

4.1 Vagues 2D

Ajouter une dimension à l'algorithme TOM n'est en soi pas très compliqué. Le premier point à adapter est celui de la propagation de la vague.

En utilisant la technique des coordonnées théoriques, la propagation de la vague en 2D est assez simple, et permet même une généralisation pour les cartes à N dimensions. La technique consiste tout simplement à reporter notre espace d'origine sur un espace à une dimension défini par la droite passant par $k_{ff}(n)$ et $k_{ff}(n-1)$ (fig. 21).

Dans un premier temps, nous cherchons à calculer les coordonnées de la source de l'interaction latérale, $\tilde{k}(n)$. Nous avons à notre disposition toutes les données nécessaires :

- les coordonnées de $k_{ff}(n-1)$, la source de la vague,
- les coordonnées de $k_{ff}(n)$, le vainqueur courant du feedforward,
- $d = isi_n.v$, la distance de propagation de la vague.
- int, la distance de déviation provoquée par l'interaction latérale, calculée avec d par le biais de la fonction d'interaction latérale.

On déduit $d' = |k_{ff}(n-1)k_{ff}(n)|$, pour finalement conclure que dans toute dimension x de la carte,

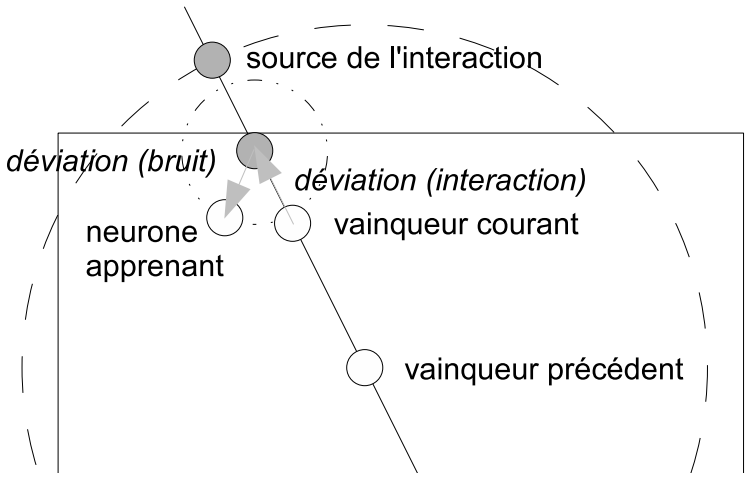


FIG. 21 – Propagation de la vague sur une carte 2D, et calcul de la position finale du neurone apprenant

$$\frac{k_{ff}(n-1)_x - k_{ff}(n)_x}{k_{ff}(n-1)_x - \tilde{k}(n)_x} = \frac{d'}{d} \tag{8}$$

A partir de cette égalité, on peut aisément calculer les coordonnées de $\tilde{k}(n)$. Lorsque ces coordonnées sont calculées, on utilise le même principe pour reporter la distance de déviation sur le vecteur $\overrightarrow{k_{ff}(n)\tilde{k}(n)}$, afin d'obtenir le vecteur de déviation final.

4.2 Bruit 2D

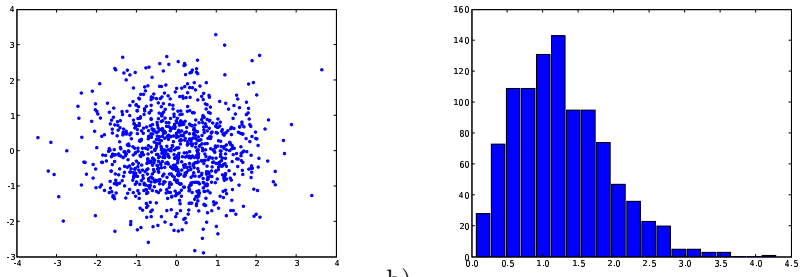
Le deuxième point à adapter pour des cartes à deux dimensions est la génération du vecteur de bruit.

Dans TOM 1D, la distribution normale centrée en 0 nous donne la distance de déviation. Le signe de la déviation nous indique sa direction. Dans un espace à deux dimensions, le bruit s'exprime sous la forme d'un vecteur de valeurs aléatoires. La difficulté consiste à générer ces valeurs aléatoires, afin que la distribution du bruit respecte les propriétés suivantes :

- la distribution de l'orientation du vecteur doit être uniforme
- la distribution de la norme du vecteur doit suivre la valeur absolue d'une loi normale $(N)(0, \sigma_{noise}(n)$

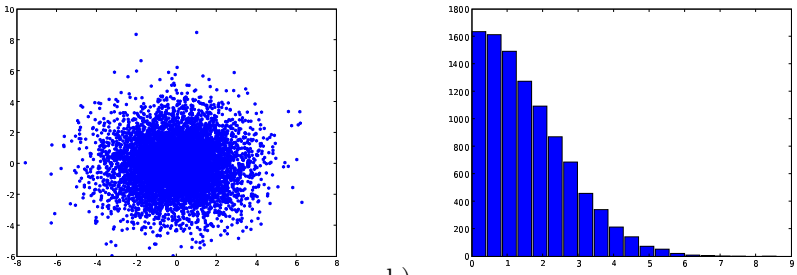
4.2.1 Inadéquation de la loi multinormale

Générons ce vecteur de bruit à l'aide d'une loi multinormale. La matrice de variance-covariance nous permet de définir la déviation de chaque composante



a) b)

FIG. 22 – (a) La distribution d'une loi multinormale aux composantes indépendantes est uniforme sur l'orientation des vecteurs. (b) La distribution de la norme ne suit pas la valeur absolue d'une loi normale centrée en 0.



a) b)

FIG. 23 – bruit 2D : 10000 points, $\sigma_{noise} = 3$. (a) distribution 2D (b) histogramme de la norme euclidienne des vecteurs bruit.

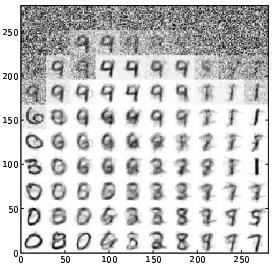
et leurs relations. Afin que la distribution de l'orientation soit uniforme, on utilise des composantes indépendantes. Malheureusement, on constate que la distribution de la norme du vecteur bruit ne répond pas à nos attentes (fig. 22).

4.2.2 Solution

Afin d'obtenir un bruit en 2D dont la norme suive une loi normale de déviation $\sigma_{noise}(n)$, nous utilisons une technique qui permet de généraliser le calcul du bruit au cas à N dimensions de manière simple et efficace :

- générer un vecteur de bruit selon une loi multinormale aux composantes indépendantes, centrées en zéro et de déviation 1 ;
- calculer $d = |\overrightarrow{noise}|$;
- générer $r = (N)(0, \sigma_{noise})$ la longueur désirée pour la norme du vecteur bruit ;
- mettre à l'échelle en faisant $\frac{\overrightarrow{noise} * r}{d}$.

a)



b)

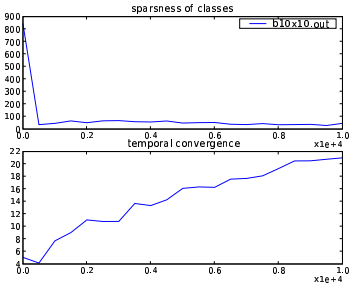


FIG. 24 – Apprentissage du jeu de données “chiffres manuscrits” sur une carte 2D de 10x10. $\sigma_{noise} = 14$, $v = 1.4$ (valeurs indexées sur la diagonale de la carte). (a) état final de l'apprentissage (b) courbe de convergence spatio-temporelle.

4.3 Auto-organisation 2D : affaiblissement de l'influence de la topologie temporelle

Nous avons vu lors de sa présentation que notre mesure fonctionne bien avec des données non-triviales dans le cas des cartes à une dimension (fig. 11). La figure 24 nous montre le résultat de l'apprentissage du même jeu de données sur une carte 2D. Le bruit a été initialisé aux dimensions de la carte, conformément aux recommandations de Wiemer. La vitesse de propagation a été initialisée en conséquence à 1.4, de telle sorte que la séquence pleinement étendue recouvre une distance égale à la diagonale de la carte.

On constate tout d'abord que l'apprentissage n'a pas recouvert toute la carte. Le nord est encore vierge et un affichage de tout l'historique de la carte révèle que l'apprentissage s'étend depuis le milieu du bord sud. Augmenter le bruit initial ne résout pas ce problème d'apprentissage partiel. La carte n'apprend totalement que lorsque le point de départ de l'apprentissage se situe au centre de la carte, et ce cas est peu probable à cause de la force du bruit initial, comme nous l'avons montré précédemment.

La seconde constatation marquante et quelque peu décevante est la courbe croissante sur la convergence temporelle. Cela signifie qu'à mesure que la carte apprend, elle respecte de moins en moins l'information temporelle contenue dans les données. Ce phénomène de croissance est associé à la propagation progressive de la zone d'apprentissage sur la carte. L'initialisation de la carte par le bruit initial ne fonctionnant pas, l'apprentissage commence par se concentrer sur une petite sous-partie de la carte. Dans cette phase, les distances dictées par la topologie temporelle ne sont pas respectées car elles ne sont pas encore atteintes. C'est pourquoi on constate une amélioration de la convergence temporelle au début, jusqu'à ce que l'étalement de la zone d'apprentissage atteigne les mesures de la topologie temporelle. Passé ce point, l'apprentissage continue de s'étendre, poussé par le bruit, et les projections s'étendent, sur-apprennent, et respectent de moins en moins la topologie temporelle.

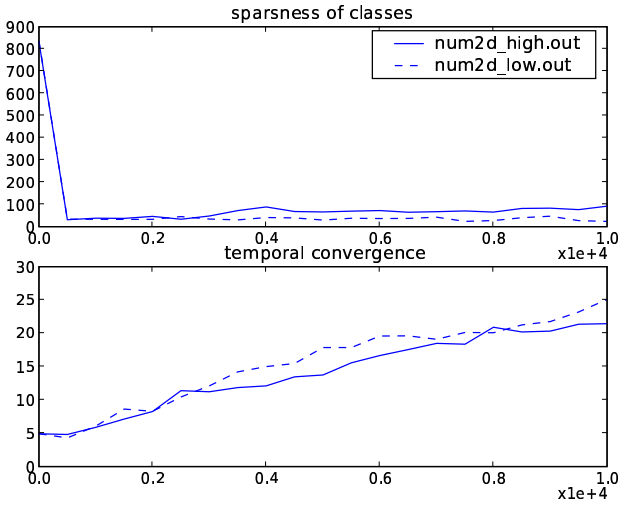


FIG. 25 – Apprentissage du jeu de données "chiffres manuscrits" sur une carte 2D de 10x10. $\sigma_{noise} = 14$, $v = 1.4$ (valeurs indexées sur la diagonale de la carte). Comparaison avec et sans interaction latérale.

Nous avons observé dans le cas 1D que les paramètres de bruit initial et de vitesse de propagation permettaient d'opérer une forme de contrôle sur l'extension de la zone d'apprentissage au sein de la classe. Cette propriété ne semble pas se retrouver dans le cas 2D, avec des paramètres similaires au cas 1D.

Malgré la perte d'influence de la topologie temporelle dans les cartes 2D, elle n'est cependant pas inexistante. En comparant la convergence de deux cartes 2D avec et sans interaction latérale, on retrouve la même situation : une convergence temporelle améliorée et une convergence spatiale dégradée pour l'apprentissage avec interaction latérale (fig. 25).

5 Implantation

Afin de tester l'algorithme TOM et ses extensions possibles, nous en avons fait une implantation.

5.1 Choix des outils

Après quelques essais (Java, OCaml), nous avons opté pour le langage Python, un langage simple et portable, bien adapté à la mise en place d'une série d'expérimentations, mais aussi puissant grâce à certaines bibliothèques.

5.1.1 SciPy

SciPy (http ://www.scipy.org/) est une bibliothèque de calcul scientifique pour python. Cette bibliothèque offre de nombreuses possibilités dans le domaine de la manipulation de matrices à N dimensions. Les calculs de notre modèle se résumant le plus souvent à des traitements sur des matrices de valeurs flottantes, l'usage de cette bibliothèque est tout à fait approrié.

Python étant un langage interprété, il n'est pas optimisé pour la performance. L'apprentissage de réseaux de neurones est une tâche lourde en calculs : les mauvaises performances de python, comparées à celles d'un programme en C/C++, risquent d'être handicapantes. Mais c'est ici que SciPy prend tout son intérêt : ses routines de calcul sont écrites en C, pour obtenir de bonnes performances. Ainsi, on a un outil qui a la puissance de calcul du C, et la facilité d'utilisation du Python.

5.1.2 Matplotlib

Un autre facteur important qui a motivé ce choix est la nécessité de visualiser les effets de l'algorithme sur l'évolution de la carte. Ceci nécessite une bibliothèque qui permet d'afficher simplement des matrices de poids, ou de tracer des courbes de suivi de la simulation.

Encore une fois, Python possède un outil avancé parfaitement adapté à cette tâche : la bibliothèque matplotlib (http ://matplotlib.sourceforge.net/). De plus, celle-çi fonctionnant de paire avec SciPy, des tâches comme le rendu graphique d'une matrice de poids sur une interface graphique ou dans un fichier postscript ne nécessitent que quelques lignes de code.

5.2 Fonctionnalités

L'implantation finale de notre programme prend la forme d'un module python. Celui-ci peut être chargé dans l'interpréteur pour construire des simulations de manière interactive, ou être importé dans un programme pour automatiser la construction.

Les fonctionnalités du programme sont les suivantes :

- parsing de fichiers textes pour construire les jeux de données ;
- sauvegarde/chargement de jeux de données construits ;
- construction de cartes à N dimensions basées sur les *ndarray* de SciPy ;
- construction de jeux de paramètres pour les simulations ;
- choix entre plusieurs méthodes de tirage de stimuli ;
- la possibilité de définir la taille de la fenêtre temporelle ;
- exécution pas-à-pas ou intégrale de simulations ;
- sauvegarde de la trace d'exécution de l'apprentissage ;
- analyse de traces d'exécution.

Pour nos travaux, nous avons écrit plusieurs programmes utilisant notre bibliothèque :

– une interface en ligne de commande nous permet de construire et exécuter des simulations, en fonction des paramètres passés au programme. Ce programme permet de visualiser l'évolution de la carte durant l'apprentissage, et de sauver la trace de l'exécution.
– un second programme en ligne de commande nous permet d'effectuer les analyses des traces d'exécution. Les mesures d'organisation spatio-temporelle sont générées à partir de N exécutions, et on affiche la moyenne des exécutions et l'écart type.

Quatrième partie
Conclusion

Le travail présenté ici montre que l'auto-organisation basée sur des données spatio-temporelles est un problème complexe, qui nécessite plus que la simple juxtaposition d'un modèle spatial et d'un modèle temporel.

L'algorithme TOM offre un modèle faisant intervenir l'information temporelle dans l'organisation spatiale d'une carte 1D de neurones. Si ce modèle ne permet pas la restitution de séquences, même simples, il propose une explication aux résultats des expériences de Spengler [12], qui pourrait être un premier pas vers un modèle de l'auto-organisation corticale universelle et constitue donc un modèle intéressant. Cependant, pour poursuivre dans cette orientation bio-inspirée, le passage que nous avons effectué aux cartes 2D, plus représentatives de la topologie corticale, était nécessaire.

Notre étude de l'algorithme TOM nous a permis de constater comment le bruit et la vitesse de propagation permettent de contrôler l'étendue de la zone d'apprentissage. Cependant, si cette possibilité existe dans le cas 1D, elle semble ne pas se retrouver dans le cas 2D. Plus précisément, l'influence de la topologie temporelle est grandement réduite dans une carte 2D. De plus, le bruit initial, censé uniformiser la carte pour permettre une bonne répartition de l'apprentissage, ne remplit pas son rôle dans les cartes 2D ; on se retrouve alors avec une diffusion lente et progressive de l'apprentissage, menant à des disparités fortes.

Par ailleurs, le principe de propagation en vague offre un mécanisme intéressant de la prise en compte de l'information temporelle dans les topographies neuronales, par le biais d'une interaction novatrice. Cependant, certains aspects vont à l'encontre de son orientation bio-inspirée. Ainsi, la présentation aléatoire des stimuli n'est pas représentative de la manière dont l'information nous parvient dans la nature. De même, les interactions temporelles rétroactives peuvent être interprétées comme un apprentissage à rebours, biologiquement peu probable. C'est pourquoi nous avons étudié le comportement de l'algorithme dans le cas d'une présentation séquentielle, plus proche des réalités biologiques. Il résulte de nos études que le mécanisme d'apprentissage utilisé par l'algorithme TOM perd en performance avec ce type de présentation. En effet, la présentation séquentielle est moins riche en interactions temporelles, ce qui a pour conséquence

de limiter l'auto-organisation temporelle a un aspect local, insuffisant pour garantir une convergence temporelle globale optimale.

Enfin, nous avons évoqué plus haut la difficulté que l'algorithme TOM éprouve à éviter des disparités d'apprentissage sur la carte, en particulier sur les cartes 2D. Ceci est pourtant un élément déterminant de la qualité du résultat. Le bruit et l'état initial de la carte étant les principaux responsables de ce phénomène, il pourrait être intéressant de les modifier. En particulier, il serait intéressant à l'avenir de remplacer l'usage du bruit par une autre forme d'organisation spatiale comme celle utilisée par SOM. De même, la mise en place d'une initialisation de la carte basée sur le jeu d'apprentissage permettrait d'obtenir une organisation mieux répartie.

Références

[1] Kohonen (T.), *Self-organizing maps*, coll. « information science ». Springer, Berlin, 3e édition, 2001.

[2] Wiemer (J.), « The time-organized map algorithm : Extending the self-organizing map to spatiotemporal signals », *Neural Computation*, vol. 15, no 5, 2003.

[3] Rabiner (L.) et Juang (B.), « An introduction to hidden markov models », *ASSP Magazine, IEEE*, vol. 3, no 1, 1986.

[4] Kremer (S.), « Spatio-temporal connectionist networks : a taxonomy and review », *Neural Computation*, vol. 13, no 6, 2001.

[5] Kremer (S.), Guilherme (A.), Barreo (A.) et Chen (D.), « A taxonomy for spatio-temporal connectionist networks revised : the unsupervised case », *Neural Computation*, vol. 15, no 6, 2003.

[6] Vaucher (G.), *A la recherche d'une algèbre neuronale spatio-temporelle*. Thèse de doctorat, université de Rennes 1, 1996.

[7] Mozayyani (N.), *Introduction d'un codage spatio-temporel dans les architectures classiques de réseaux de neurones artificiels : application à la reconnaissance des caractères manuscrits*. Thèse de doctorat, université de Rennes 1, 1998.

[8] Neji Ben Salem (Z.). « Auto-organisation spatio-temporelle : application à la reconnaissance de parole ». université de Tunis, thèse en cours.

[9] Bennani (Y.), Zehraoui (F.) et Fessant (F.), *Apprentissage connexionniste*, chap. 7. 2006.

[10] Thorpe (S.), « L'efficacité du traitement visuel humain : implications pour la computation dans le système visuel. », 1992.

[11] Wiemer (J.), *Learning topography in neural networks : towards a better understanding of cortical topography*. Thèse de doctorat, université de Bochum, 2000.

[12] Spengler (F.), Hilger (T.), Wang (X.) et Merzenich (M.), « Learning induced formation of cortical populations involved in tactile object recognition », *Soc Neurosc Abstracts*, vol. 22, no 1, 1996.

[13] Geldard (F.) et Sherrick (C.), « The cutaneous "rabbit" : a perceptual illusion », *Science*, 1972.

[14] Lo (Z.) et Bavarian (B.), « On the rate of convergence in topology preserving neural networks », *Biol Cybern*, 1991.

[15] Schmajuk (N.), *Animal Learning and Cognition : A Neural Network Approach*. 1997.